

# Lab0 Report

---

## Phase 0~3

按照教程视频的思路即可解决。

## Phase 4

Phase 4 要求输入一个字符串，该字符串会经过两次加密操作，最后炸弹会检查加密后的字符串与预期字符串是否匹配。

对于一阶段的加密，我的思路是在第二次加密前打上断点，使用 gdb 检查寄存器 `x0` 中储存的值。经过几次实验就可以发现，输入字符串的奇数位会被排到偶数位之前，比如 `abcd` 会被加密为 `acbd`。

对于二阶段的加密，由于加密代码看不懂，并且无法找出明显的规律，我尝试了所有的字母输入，并将它们对应的转换后字母记录下来。然后根据汇编代码检查在 `0x4a000+#104` 地址储存的字符串，然后根据先前记录的转换表得到二阶段期望输入，最后再得到一阶段的期望输入。

## Phase 5

Phase 5 涉及一个二叉树，输入的数字会和每个节点的数字大小进行比较，大则走右节点，小则走左节点。

具体逻辑是，输入一个数  $x$ ，以及二叉树的根节点的值  $y$ 。将  $x$  作为 `func_5` 的输入，比较  $x$  与  $y$ ，若大于，则  $y$  的值变为根节点右节点的值，再次执行 `func_5`，随后执行  $z = 2z + 1$ ， $z$  为输出。若小与，最后执行  $z = 2z$ 。当访问的子节点为 0 时，令  $z = 0$ 。在递归执行完 `func_5` 后，输出  $z = 3$  则炸弹拆除。

通过 gdb 检查二叉树可以发现这是一个 4 层二叉树，也就是说我们要找到一个从叶子到根节点的路径，使得输出 0 经过三次操作后等于 3。经过简单的推理，分别是  $0 \times 2 = 0$ ,  $0 \times 2 + 1 = 1$ ,  $1 \times 2 + 1 = 3$ 。因此路径是根节点、右节点，右节点，左节点（叶子）。因此只需要保证输入的数字大于前三个节点的值，小于最后一个节点的值即可